This is a preprint of an article published in International Review of Electrical Engineering (I.R.E.E.), Vol. 6, N. 3 May-June 2011

A New Preconditioner Design Based on Spectral Division for Power Flow Analysis

Hasan Dağ[1], E. Fatih Yetkin[2], Murat Manguoğlu[3]

Abstract – *Solution of large sparse linear systems is the most time consuming part in many power system simulations. Direct solvers based on LU factorization, although robust, are known to have limited scalability on parallel platforms. Thus, Krylov subspace based iterative methods (i.e. Conjugate Gradient method, Generalized Minimal Residuals (GMRES) method) can be used as alternatives. To achieve competitive performance and robustness, however, the Krylov subspace methods need a suitable preconditioner. In this work we propose a new preconditioner for iterative methods, which can be used in Newton-Raphson process of power flow analysis. The suggested preconditioner employs the basic spectral divide and conquer methods and invariant subspaces for clustering the eigenvalues of the Jacobian matrix appearing in Newton-Raphson steps of power flow simulation. To obtain the preconditioner, we use Matrix Sign Function (MSF) and to obtain the MSF itself we use Sparse Approximate Inverse (SPAI) algorithm with Newton iteration. We compare the convergence characteristics of our preconditioner against the well-known black-box preconditioners such as incomplete-LU and SPAI.*

Keywords: *Iterative Methods, Power Flow Analysis, Spectral Projectors, Preconditioner*

Symbol list:

$\boldsymbol{A}$	nxn coefficient matrix
$\boldsymbol{B}$	nx1 right hand side vector
$\boldsymbol{X}$	solution vector
$\Lambda(A)$	spectrum of matrix $\boldsymbol{A}$
$\boldsymbol{P}$	nxn projector matrix
$\boldsymbol{I}$	nxn identity matrix
$\boldsymbol{Q_b}$	nxn orthogonal matrix
$\boldsymbol{M_b}$	nxn preconditioner matrix
$\boldsymbol{M}$	sparse approximate inverse of the matrix $\boldsymbol{A}$
sign($\boldsymbol{A}$)	Matrix Sign Function of matrix $\boldsymbol{A}$
nnz(**A**)	Number of non-zeros of matrix $\boldsymbol{A}$
ρ ,η	the number of positive and negative eigenvalues of matrix $\boldsymbol{A}$, respectively.
α , β	Real numbers to determine the borders for eigenvalues

I. Introduction

The power flow problem in an electrical transmission system is mainly based on a solution of a set of non-linear equation. Application of the Newton-Raphson method to the non-linear set of equations results at each Newton-Raphson iteration in a linear system and has the form below.

$$\begin{bmatrix} H & N \\ M & L \end{bmatrix} \begin{bmatrix} \Delta\theta \\ \Delta V \end{bmatrix} = \begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} \quad (1)$$

These equations represent only one step of the power flow problem. Several applications and solution methods about power flow analysis can be found in the literature [1-5]. They are traditionally solved by the direct methods with sparse techniques [1]. This equation set is in the form of $\boldsymbol{Ax=b}$ and this notation will be used for the rest of the paper.

Much attention has been paid to the solution of linear systems. While direct solvers are robust methods [6], iterative methods have also been shown to be almost as robust as direct solvers if a suitable preconditioner is used. Richardson iterations, Krylov subspace methods, and Chebyshev iterations are some examples of the most commonly used iterative methods. These types of iterative methods can be classified in two different subclasses as symmetric and un-symmetric methods according to the type of the coefficient matrices. For large and sparse systems, Krylov subspace methods (such as CG, GMRES, etc.) have been proven to be much more effective. Their advantage over direct methods [7], [8] is two folds: (i) better parallelism (ii) less memory requirement. Much work has been done on preconditioning techniques [9-10].

The preconditioner we propose aims to remove extreme eigenvalues of the Jacobian matrix with the help of an orthogonal similarity transformation. The method only needs some basic information about the eigenvalue spectrum of matrix $\boldsymbol{A}$. This information can easily be obtained by the eigenvalue inclusion theorems, such as that of Gerschgorin or the well-known Power iteration

Manuscript received February 2011, revssed.March 2011

[11]. With this information some regions can be defined in the complex plane to construct the orthogonal transformation matrices to remove the extreme eigenvalues. It is proved that the eigenvalues of the Jacobian matrix in the power flow problem do not change significantly in Newton-Raphson steps [1]. One of the advantages of the proposed preconditioner is based on this fact. The core computational effort needed is, thus, in the first step of the Newton-Raphson iteration since the same preconditioner can be used in the subsequent Newton-Raphson iterations. The basic tool for this purpose is matrix sign function (MSF), which behaves like its scalar equivalent. Scalar sign function, extracts the sign of a real number. Similarly, MSF detects the signs of eigenvalues of a matrix and it returns two blocks of identity matrix. The size of the first block of identity matrix gives us the number of positive eigenvalues of the matrix and the size of the second one gives the number of negative eigenvalues. Furthermore, one can perform a rank-revealing QR decomposition on the $sign(\boldsymbol{A})+\boldsymbol{I}$ or $sign(\boldsymbol{A})-\boldsymbol{I}$ to compute an orthonormal basis for the invariant subspace of the eigenvectors in the right or the left half-plane [12]. In addition to that one can implement the same operations on a shifted matrix to find the number of eigenvalues larger or smaller than a selected real number or the invariant subspace of the eigenvectors on a desired part of the complex plane. This orthonormal base can be employed to remove the extreme eigenvalues of the Jacobian matrix. In fact, the proposed method uses more floating-point operations than those of the similar type preconditioners. On the other hand, the method is easy to parallelize and the building blocks of the algorithm are all well-known block matrix operations like matrix multiplication or QR decomposition.

Most time consuming part of the suggested preconditioner is the computation of MSF. It requires matrix inversion. The Sparse Approximate Inverse (SPAI), however, can be used to obtain an approximate MSF, which can be used in the construction phase of the preconditioner. SPAI technique is an efficient preconditioner implementation for sparse linear systems [13].

In this work, we especially focus on developing the method. Therefore, well-known IEEE test cases (30 bus, 57 bus, 118 bus and 300 bus) are used to show the accuracy and the effectiveness.

The rest of the paper is organized as follows. In the second section, mathematical tools of the algorithm are briefly introduced and the algorithm itself is described. In the third section, some numerical test results and comparisons are given. Finally, in the last section some conclusions and future work are presented.

II. Method and Algorithm

A new preconditioner based on matrix sign function (MSF) and spectral decomposition is presented. The aim of a preconditioner can be thought as the process of grouping of the eigenvalues of the coefficient matrix at hand. In the proposed method, MSF is employed to group the eigenvalues of the coefficient matrix $\boldsymbol{A}$. MSF is a very powerful and useful tool for matrix analysis. It is possible to employ the MSF to build a spectral projector [14].

Definition 2.1: Let $\Lambda(Z)$ show the spectrum of *nxn* dimensional square matrix $\boldsymbol{Z}$ and $\Lambda(Z) = \Lambda_1 \cup \Lambda_2$, $\Lambda_1 \cap \Lambda_2 = \emptyset$. If $\boldsymbol{P}$ is the invariant subspace of $\Lambda_1(Z)$ any projector onto $\boldsymbol{P}$ is called as spectral projector. Basic properties of the spectral projectors can be given as:

- $range(\boldsymbol{P})=range(\boldsymbol{AP})$
- $ker(\boldsymbol{P})=range(\boldsymbol{I-P})$, $range(\boldsymbol{P})=ker(\boldsymbol{I-P})$
- $(\boldsymbol{I-P})$ is a spectral projector for $\Lambda_2(Z)$

Spectral projectors can be used to split the matrices into diagonal blocks based on its eigenvalues. The algorithm for block decomposition of a matrix by spectral projectors is given in its general form in Alg. 1 [14].

Algorithm 1 BLOCK DECOMPOSTION

Input: Spectral projector $\boldsymbol{P}$

Output: Diagonal blocks of matrix $\boldsymbol{A}$

1. Compute rank revealing **QR** decomposition of the projector as $P = QR\Pi$.
2. Build $\boldsymbol{A}$-invariant S_1 subspace from the first k column of the orthogonal matrix **Q**.
3. Compute the below transformation

$$\hat{A} = Q^T A Q = \begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix} \tag{2}$$

4. $\Lambda(A_{11}) = \Lambda_1$ and $\Lambda(A_{22}) = \Lambda_2$

II.1. Computation of Matrix Sign Function

There are a few ways to compute MSF in the literature [15]. One of the main approaches of MSF computation is based on Newton iteration, which uses the equation below.

$$S^2 = I \tag{3}$$

Here, $S = sign(A)$, is a sign function of matrix $\boldsymbol{A}$ and $\boldsymbol{I}$ is an identity matrix in appropriate dimensions. If the Newton iteration method is applied to (3) one can obtain the iteration for MSF.

$$A_{i+1} = \frac{1}{2}(A_i + A_i^{-1}) \quad with \quad A_0 = A \tag{4}$$

Iteration in (4) is quadratically convergent in global sense [12]. But it uses the inverse of **A**, which makes (4) computationally expensive. There are many variations of this iteration and they can be found in the literature [16], [17], [18]. But most of these methods are also based on matrix inversion and hence they are not suitable to be used as a preconditioner. Obviously, if the inverse of the

matrix $\boldsymbol{A}$ is available, a matrix-vector product can solve the equation set.

To overcome this problem, we combine some special sparse matrix tools like Sparse Approximate Inverse (SPAI) and sparse QR decomposition to compute matrix sign function decomposition in shorter time with a satisfactory accuracy. SPAI approach itself can also be used as a preconditioner. Some comparisons are also made between SPAI preconditioners and our proposed preconditioner.

SPAI [19] algorithm computes a sparse approximate inverse $\boldsymbol{M}$ of a given sparse matrix $\boldsymbol{A}$ by minimizing $\| AM - I \|_F$ provided that the sparsity structure of $\boldsymbol{M}$ is the same as the original matrix.

To compute MSF of a matrix $\boldsymbol{A}$, one can employ the SPAI method to produce the approximate inverse in every step of the Newton iteration given in (4). Because of the use of approximate inverse instead of exact inverse, the number of Newton iterations in (4) increases.

Another important advantage of using SPAI to obtain MSF is the preservation of sparsity. Electrical power flow Jacobian matrices mostly have very sparse structures. But if one employs the Newton iterations method, because of the matrix inversion, sparsity of the matrices will be lost. Clearly, computing the exact inverse is both computationally and storage wise expensive.

In following experiments a sparsity and accuracy comparison is performed between the MSF matrices computed by direct inversion in (4) and SPAI.

TABLE I
COMPARISON OF THE MATRIX SIGN FUNCTIONS OBTAINED BY SPAI AND DI (DIRECT INVERSE)

Test Case	Number of Iterations		Sparsity		Accuracy	
	SPAI	DI	SPAI	DI	SPAI	DI
IEEE57	86	10	0.099	1	$6.8*10^{-10}$	$4.6*10^{-15}$
IEEE118	81	8	0.036	1	$4.2*10^{-10}$	$1.5*10^{-14}$
IEEE300	42	10	0.014	1	$1.5*10^{-10}$	$1.8*10^{-15}$

In all experiments the tolerance for Newton iterations is 10^{-10}. In the Table I, two different measures are used to compare the different computational techniques of MSF. Sparsity is defined as the ratio of non-zero elements to square of dimension of the matrix. It can be defined as,

$$sparsity = \frac{nnz(A)}{n^2} \tag{5}$$

where n is the dimension and nnz stands for the number of non-zeros of $\boldsymbol{A}$. The accuracy of the computation is given by,

$$accuracy = norm(sign(A)^2 - I) \tag{6}$$

where $\boldsymbol{I}$ is the identity matrix. The convergence graph for both type of computations are given in Fig. 1.

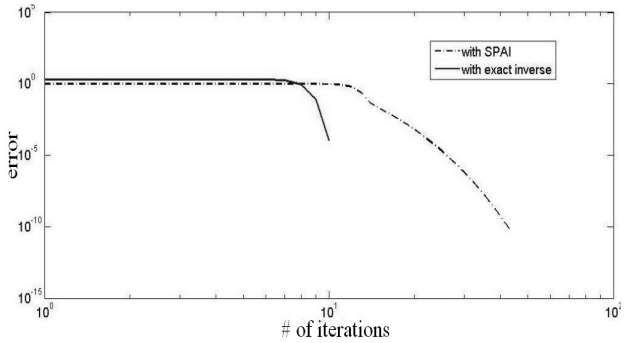

Fig. 1. Convergence history of MSF for IEEE 300 bus system, when SPAI and exact inverse methods are used in (4).

As shown in Table I and Fig. 1, if SPAI is used to obtain the approximate inverse, number of iterations of the Newton method for MSF increases. This increase, however, can be compensated by the main advantage of using SPAI in MSF computation, namely, the preservation of the sparsity structure and the reduced computational cost. The sparsity preservation is important to obtain satisfactory computational cost for the preconditioner preparation step of the whole solution procedure. In Fig. 2 sparsity structure of the MSF for Jacobian matrix of the IEEE 300 bus test system is given.

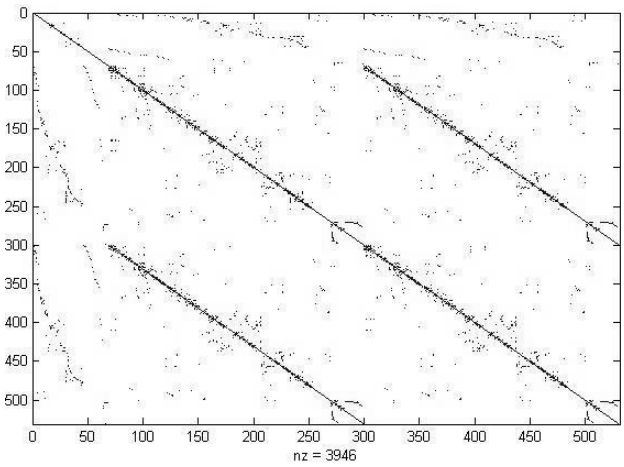

Fig. 2. Sparsity structure of the sign(A) for IEEE 300-bus test system when SPAI is used for inversions. Here, the sparsity structure of **A** is preserved as much as possible by using proper parameters in SPAI algorithm.

II.2. Counting Eigenvalues with Matrix Sign Function

To build a spectral projector effectively, some information regarding the spectral of $\boldsymbol{A}$ is needed. The number of the eigenvalues in slices of complex domain can give us a rough idea about the distribution of the eigenvalues. Although there are some methods for counting eigenvalues using characteristic polynomials, such as Gleyse, Wilf, methods etc., they are not computationally feasible [20]. Instead, MSF can be used for counting the eigenvalues. To compute the numbers of eigenvalues in a pre-determined slice of complex plane

one can use the basic properties of the matrix sign function [14].

Theorem 2.1: Let ρ, η be the numbers of the positive and negative eigenvalues of matrix $\boldsymbol{A}$, respectively. Then the trace of $sign(\boldsymbol{A})$ can be computed by $trace(sign(\boldsymbol{A}))= \rho - \eta$. In addition, the size of the matrix $\boldsymbol{A}$ is equal to $n = \rho + \eta$. From this point, one can obtain the relationships below.

$$\rho = \frac{1}{2}(n + trace(sign(A)))$$
$$\eta = \frac{1}{2}(n - trace(sign(A))) \quad (7)$$

Above theorem gives the number of eigenvalues with respect to the origin. If the origin is shifted with some scalar β, it can still be used to obtain the number of the eigenvalues larger or smaller than β.

Theorem 2.2: Let $\beta \in \Re$ and assume that there is no eigenvalues of $\boldsymbol{A}$ with real part equal to β. Let ρ, η denote the number of the eigenvalues with real parts larger and smaller than β, respectively. Then, we have

$$\rho = \frac{1}{2}(n + trace(sign(A - \beta I)))$$
$$\eta = n - \rho \quad (8)$$

This approach can be used to determine the number of the eigenvalues of matrix $\boldsymbol{A}$ in $[\alpha, \beta]$ slice in the complex plane [14].

Theorem 2.3: Let $\alpha, \beta \in \Re$ and assume that there are no eigenvalues of matrix $\boldsymbol{A}$ with real parts neither equal to α nor to β. In that case, one can find ρ that shows the number of the eigenvalues of matrix $\boldsymbol{A}$ with real parts in between α and β as;

$$\rho = \frac{1}{2}(trace(sign(A - \alpha I) - sign(A - \beta I))) \quad (9)$$

Gerschgorin theorem can be used to specify the largest borders for the eigenvalue spectrum of matrix $\boldsymbol{A}$. Then some linear slicing can be used to determine the distribution of the eigenvalues with MSF. This idea is illustrated in Fig. 3. Another possible solution for the selection of a border for eigenvalue distribution is the power iteration [11]. Using power iteration the real part of the largest eigenvalue can be found approximately and then it can be used as the border value. The importance of the determination of the border value is about the selection of β. Once the border is selected it is possible to find an appropriate β as a ratio of the border value.

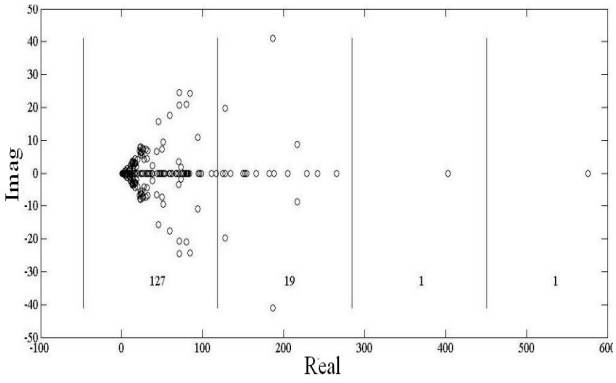

Fig. 3. The number of the eigenvalues for the Jacobian of the 118-bus test system in the first step of the Newton-Raphson algorithm. This graph is obtained with MSF using 5 different β values. In the figure, eigenvalues of IEEE 118-bus test case clustered in between the origin and 110.

The selection of the β parameter is completely problem dependent. But in the experiments we have observed that 30-40% of real part of the largest eigenvalue will be enough for an accurate and reliable preconditioner. The most important point in this estimation is the fact that the percentage is not related to the number of eigenvalues. In other words, the selected domain does not need to cover the same percentage of the eigenvalues. In all of the following experiments β is chosen as 40% of real part of the maximal eigenvalue.

II.3. Computing Invariant Subspaces via MSF

MSF can be employed to compute a matrix whose eigenvalues are equal to the eigenvalues of matrix $\boldsymbol{A}$ in a specific range [14]. More technically:

Theorem 2.4: Let $\beta \in \Re$ and matrix **S** be defined as:

$$S = \frac{1}{2}(I + sign(A - \beta I)) \quad (10)$$

By applying the rank revealing QR onto this matrix $\boldsymbol{S}$,

$$Q_b^T S\Pi = \begin{bmatrix} S_{11} & S_{12} \\ 0 & 0 \end{bmatrix} \quad (11)$$

is obtained. Here, $\boldsymbol{S}_{11}$ is a kxk dimension matrix and k equals to the number of the eigenvalues of matrix $\boldsymbol{A}$ that are larger than β. The orthogonal matrix $\boldsymbol{Q}_b$ can be used for the transformation below.

$$B = Q_b^T A Q_b = \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} \quad (12)$$

Finally, matrix $\boldsymbol{B}_{11}$ is a kxk matrix whose eigenvalues are equal to the eigenvalues of matrix $\boldsymbol{A}$ with real parts larger than β. Since $\boldsymbol{S}$ is a sparse matrix, one can use appropriate sparse QR decomposition methods.

Therefore, it is possible to obtain $\boldsymbol{B}$ without computing $Q^T AQ$ explicitly.

The same procedure in eigenvalue counting algorithm can be used to obtain the invariant subspaces in above theorem and it is possible to compute different $\boldsymbol{B_{11}}$ matrices based on the geometry selection in the complex plane.

II.4. Building a Preconditioner with MSF

To accelerate the convergence rate of an iterative method for solution of linear systems, a preconditioner has to be used. The preconditioners are thoroughly studied in the literature [9], [21]. We shall keep in mind that there is not a single preconditioner that is suitable for all types of linear systems. Preconditioning mainly aims to reduce the condition number of the coefficient matrix. Basically the process can be summarized as reducing the groups of the eigenvalues of the coefficient matrix. In this work, orthogonal spectral projectors are employed to reorder and to deflate the effect of the extreme eigenvalues of the coefficient matrix $\boldsymbol{A}$. Based on Theorem 2.4; one can observe that MSF can also be useful for this purpose.

In the proposed method, MSF is used to produce an orthogonal projector for transforming matrix $\boldsymbol{A}$ into the form given in (12). After that operation one can define several preconditioners to accelerate the iterative method. In this work, we use GMRES. In [22], MSF is explained in detail. In this work, three different approaches with different computational tools (like SPAI method and power iteration) are investigated for the preconditioner design given in [22].

1) First Approach: After a suitable β value is found with the help of either the power iteration or Gerschgorin discs, Theorem 2.4 can be used to obtain the block diagonal form of the matrix $\boldsymbol{A}$ as $\boldsymbol{B}$. Theoretically, the eigenvalues of the matrix $\boldsymbol{B_{11}}$ are equal to the eigenvalues of the matrix $\boldsymbol{A}$ whose real parts are greater than β. When SPAI method is used, however, the eigenvalues of $\boldsymbol{B_{11}}$ are no longer equal to the eigenvalues of A but they are approximately equal and this is usually acceptable for a preconditioner. The eigenvalue distributions are shown in Fig. 5. Table I shows the results of a comparison of the accuracy of the matrix sign function obtained using SPAI and exact inverse. Once orthogonal transformation matrix $\boldsymbol{Q_b}$ is computed by theorem 2.4, the preconditioner matrix $\boldsymbol{M_b}$ is obtained as a combination of an identity matrix and $\boldsymbol{B_{11}}$ matrices as given in Fig. 4.

More precisely, the preconditioner matrix for the first approach can be given as,

$$M_b = \begin{bmatrix} B_{11} & 0 \\ 0 & I_{n-k} \end{bmatrix} \qquad (13)$$

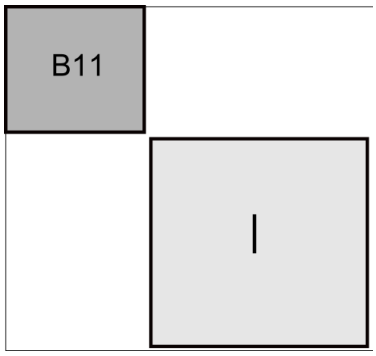

Fig. 4. Structure of the preconditioner matrix for the first approach.

where $\boldsymbol{B_{11}}$ is a *kxk* matrix and $\boldsymbol{I_{n-k}}$ is an *(n-k)*x*(n-k)* identity matrix. $\boldsymbol{B_{11}}$ contains the approximate eigenvalues of the matrix $\boldsymbol{A}$ larger than the pre-selected value β. This situation is illustrated for 118-bus IEEE test example, Here, $\beta = 80$ and it is clear that eigenvalues of $\boldsymbol{B_{11}}$ are approximate and some unexpected values are also computed by MSF with SPAI. Here SPAI is employed to preserve the sparsity structure of the matrix $\boldsymbol{A}$.
After the $\boldsymbol{Q_b}$ matrix is applied onto the linear equation set $\boldsymbol{Ax=b}$, we get

$$Q_b^T AQ_b (Q_b^T x) = Q_b^T b$$

(14) In (14), the matrix $\boldsymbol{Q_b}$ is multiplied with $\boldsymbol{A}$ from both sides to make the eigenvalues of $\boldsymbol{A}$ to be in the same

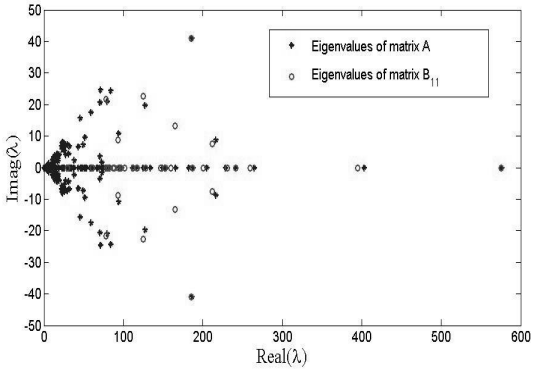

Fig. 5. Eigenvalues of the Jacobian matrix for one-step in power flow Newton-Raphson iteration and the eigenvalues of $\mathbf{B_{11}}$, which is produced by MSF with SPAI. Due to the error coming from computation of MSF with SPAI, eigenvalues of $\mathbf{B_{11}}$ are not the same as those of **A**.

order as those of the preconditioner matrix $\boldsymbol{M_b}$. Finally, the original system is transformed into $\boldsymbol{A_n x_n = b_n}$ where,

$$\begin{aligned} A_n &= Q_b^T AQ_b \\ b_n &= Q_b^T b \\ x_n &= Q_b^T x \end{aligned} \qquad (15)$$

The new linear system can be solved by any suitable iterative method with a preconditioner $\boldsymbol{M_b}$. In this case, dimension of the matrix $\boldsymbol{B_{11}}$ is determined by the pre-selected β values. In Fig. 6, dimension of the matrix $\boldsymbol{B_{11}}$

as we vary β values are shown with exact inverse and SPAI in MSF computation step of the algorithm.

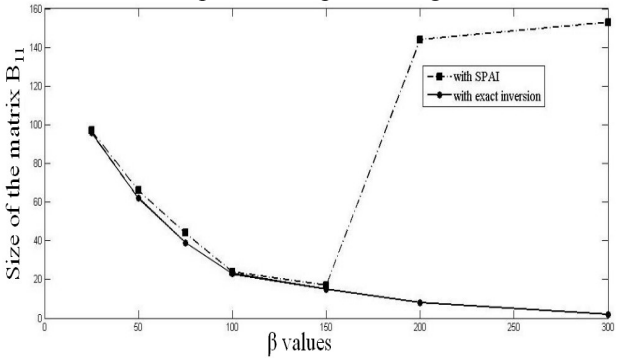

Fig. 6. Change of the dimension of B_{11} according to pre-selected β values with exact and sparse approximate inverse usage in MSF computation.

It can be seen from Fig. 6 that the size of $\boldsymbol{B_{11}}$ block in the preconditioner is mainly based on the pre-selected β value and the accuracy of the method is getting worse for larger β values. The main reason is that a larger β value causes less eigenvalues to be selected and therefore one has to use more accurate methods like exact inverses in MSF computation. The dependency of the first method to parameter β can be diminished by including $\boldsymbol{B_{22}}$ into the preconditioner. This idea is described in detail in the second approach.

2) The Second and Third Approaches (M_2 and M_3): To reduce the effect of β and to build a more reliable preconditioner one can select both of $\boldsymbol{B_{11}}$ and $\boldsymbol{B_{22}}$ blocks. In that case, the dimension of $\boldsymbol{B_{11}}$, depending on this β parameter, does not affect the accuracy and the reliability of the preconditioner based on the experimental observations. By setting dimensions of $\boldsymbol{B_{11}}$ and $\boldsymbol{B_{22}}$ to be almost equal, we can achieve better load balancing for parallelism. The structure of the preconditioner matrix in the second approach is given in Fig. 7.

Fig. 7. Structure of the preconditioner matrix for the second approach.

Implementation of the preconditioner is exactly the same as that of the first approach. Here, the effect of $\boldsymbol{B_{12}}$ is neglected since its Frobenious norm is relatively small. In order to have a basis for comparison one can come up with a third approach by including the $\boldsymbol{B_{12}}$ block into the preconditioner as well. Structure of the third type of preconditioner is given in Fig. 8.

Fig. 8. Structure of the preconditioner matrix for the third approach.

The algorithm for forming the preconditioner and solving the systems using the preconditioner are given in Alg.2. We emphasize, however, it is not required to form the preconditioner explicitly and this information is provided here only for illustration of the method.

Algorithm 2 MSF-BASED PRECONDITIONERS

Input: Matrix **A**, right hand side vector *b*, *r (percentage of eigenvalue with the largest real part)*, preconditioner type *b*.

Output: (i) Preconditioner $\mathbf{M_b}$ and (ii) Solution of the systems using the preconditioner $\mathbf{M_b}$

(i) Obtaining the preconditioner:

1. Find the eigenvalue of matrix **A** with the largest real part and assign to η .
2. Select $\beta = \eta - r\eta$.
3. Use theorem 2.4 to compute $\mathbf{B_{11}}$, $\mathbf{B_{12}}$, $\mathbf{B_{22}}$, and $\mathbf{Q_b}$ by using SPAI.
4. Build $\mathbf{M_b}$ matrices as

$$\text{If } (b = 1) \qquad M_b = \begin{bmatrix} B_{11} & 0 \\ 0 & I_{n-k} \end{bmatrix}$$

$$\text{If } (b = 2) \qquad M_b = \begin{bmatrix} B_{11} & 0 \\ 0 & B_{22} \end{bmatrix}$$

$$\text{If } (b = 3) \qquad M_b = \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix}$$

(ii) Solving systems using the preconditioner $\mathbf{M_b}$

1. Compute $\mathbf{A_n}$ and b_n using (15).
2. Use an iterative method to solve linear equations with the preconditioner $\mathbf{M_b}$ and obtain x_n
3. Retrieve the true solution $x = \mathbf{Q_b} x_n$

II.5. Numerical Results

II.6. Comparison of MSF based methods

Some popular and well-known IEEE power test systems are used for the experiments. In every step of the Newton-Raphson iteration a Jacobian matrix is created. The first Jacobian is used in the following experiments. It has been proven that the eigenvalues of the Jacobian matrix in different steps do not vary dramatically [5]. This is an important advantage of the proposed preconditioner. The preconditioner is created at the very beginning, as a preprocessing step and is used for different Jacobians. The main properties of the test matrices are given in Table II.

TABLE II
SOME NUMERICAL PROPERTIES OF THE IEEE TEST CASES

Number of buses	Matrix Size	Number of Non-zeros	Cond(A)
30	53	333	492.81
57	106	718	825.09
118	181	1051	3.17×10^3
300	530	3736	1.16×10^5

In this work, three variations of MSF-based preconditioners are proposed. In the first case, we take only the $\boldsymbol{B_{11}}$ block to form the preconditioner. However, this is not enough to build a sufficiently effective preconditioner. $\boldsymbol{B_{22}}$ and $\boldsymbol{B_{12}}$ can also be included to form the second and the third types of preconditioners. First, the proposed preconditioners are compared to each other based on their effects on eigenvalue distribution of the preconditioned system. Then, all of them are compared to the well-known Incomplete LU (ILU) methods and Sparse Approximate Inverse (SPAI) type preconditioners. In the first test, IEEE 118 bus test case is used and eigenvalue distributions of different cases are investigated. For all type of preconditioners β is selected as 80. It can be seen from the Fig. 9, the eigenvalues of the preconditioned system matrices are much better clustered around 1 for second and third approaches.
Though the eigenvalues of the preconditioned system with first approach is also clustered, still they are scattered in a relatively wide area of complex plane.

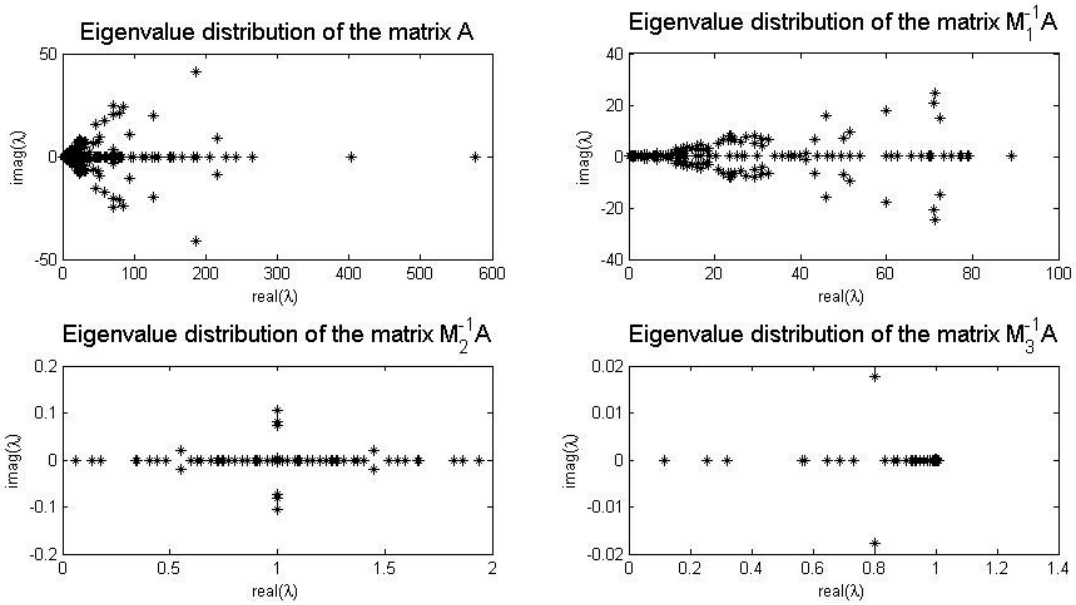

Fig. 9. The eigenvalue distributions of the matrices ***A, $M_1^{-1}A$, $M_2^{-1}A$, and $M_3^{-1}A$.*** The plots have different scales. But it can be seen from the subplots that eigenvalues with the second and the third approaches are much better clustered around 1.

II.7. Comparison between ILU, SPAI and MSF based methods

The same power flow data are used for the comparison of the most widely used preconditioning techniques, Incomplete LU (ILU), and the suggested MSF-based methods. Stand alone SPAI preconditioner is also used in the experiments for comparison. ILU methods produce an approximation for the classical LU decomposition and these incomplete ***L*** and ***U*** factors are used as preconditioner [21]. ILU factorization has several types.

1. *ILU-Threshold*: The entries of ***L*** and ***U*** matrices below some threshold value are discarded and the resultant factors are used as preconditioners.
2. *ILU('x'):* Dropping of fill-ins are decided by the sparsity pattern of matrix ***A***. For example ILU(0) means no fill-in allowed outside the sparsity pattern of matrix **A**.

The residual history of restarted GMRES algorithm with several types of ILU, SPAI, and MSF based preconditioners is shown in Fig. 10 In all tests, the restart value is chosen as 5, stopping tolerance of GMRES is chosen as 10^{-6} and maximum iteration number equals the dimension of the Jacobian matrix. Parameters for SPAI method are selected to preserve sparsity of the original matrix for both MSF methods and SPAI itself as a standalone preconditioner.

Finally, preconditioners are compared in the Newton-Raphson iterations of power flow simulation. To do this, Matpower package is used [23]. In Matpower package, the default solver is classical LU method. To test our preconditioner, the solver is replaced with GMRES obtained from the templates of NETLIB [24]. In our tests we used 118 and 300 bus classical IEEE examples, important properties of which are given in Table I.

The second and the third approaches are employed for GMRES iterations in Newton-Raphson power flow analysis. The preconditioner matrix is created only once and it used in all other Newton-Raphson steps. For IEEE-300 test case, we observed that, GMRES with ILU(0) and ILUT (0.1) does not converge to the correct value. As a result one can say that satisfactory accelerations with the suggested MSF based preconditioners are obtained.

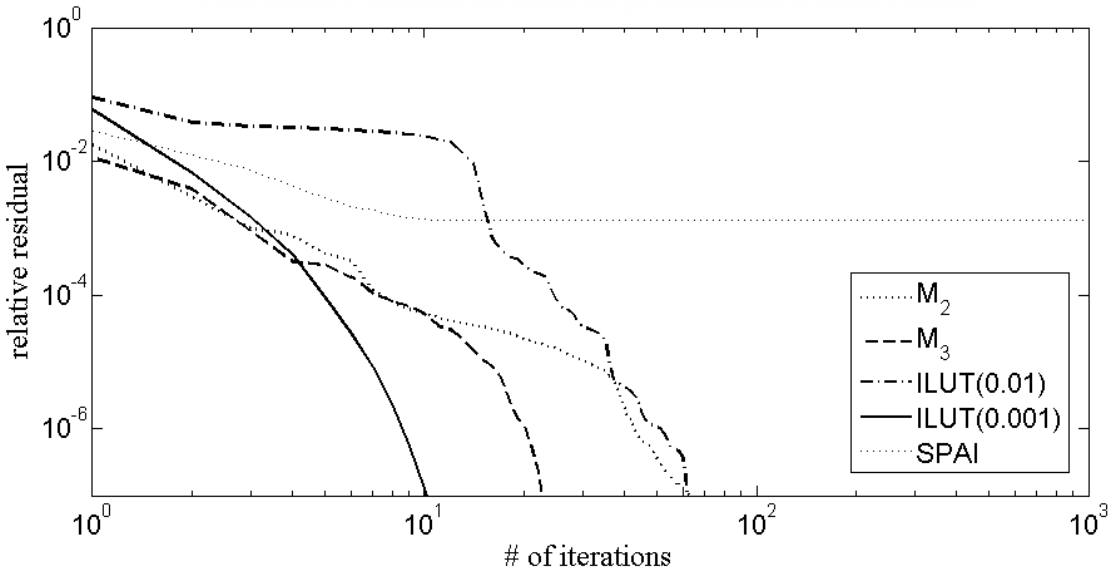

Fig. 10. The residual histories for the first Jacobian of IEEE-300 test case with several preconditioners are given. Here the second and the third approaches are in between the ILUT(0.01) and ILUT(0.001) preconditioners. So, the ILUT(0.001) preconditioner is very close to the direct solution and thus it can be said that the suggested approaches accelerate the GMRES iteration sufficiently. On the other hand if SPAI preconditioner alone used with default parameters (sparsity preserved) as in MSF computations for second and third approach, it was not converged.

TABLE III
ITERATION NUMBERS IN EACH NEWTON-RAPHSON STEP WITH A NEW PRECONDITIONER FOR EVERY STEP.
THE TEST CASE IS: IEEE – 118

NR-iteration	M_2	M_3	SPAI	ILU(0)	ILUT(0.1)	ILUT(0.01)
1	25	12	**F**	8	9	2
2	35	15	**F**	11	12	2
3	49	24	**F**	16	17	3

TABLE IV
ITERATION NUMBERS IN EACH NEWTON-RAPHSON STEP, WITH A NEW PRECONDITIONER FOR EVERY STEP.
THE TEST CASE IS: IEEE –300

NR-iteration	M_2	M_3	SPAI	ILU(0)	ILUT(0.1)	ILUT(0.01)
1	50	23	**F**	**F**	406	27
2	48	18	**F**	**F**	362	60
3	72	21	**F**	**F**	**F**	74
4	98	43	**F**	**F**	**F**	94
5	158	52	**F**	**F**	**F**	143

TABLE V
ITERATION NUMBERS IN EACH NEWTON-RAPHSON STEP WITH THE SAME PRECONDITIONER FOR EVERY STEP.
THE TEST CASE IS: IEEE–118

NR-iteration	M_2	M_3	SPAI	ILU(0)	ILUT(0.1)	ILUT(0.01)
1	28	11	**F**	8	9	2
2	41	17	**F**	13	11	2
3	58	58	**F**	17	16	2

TABLE VI
ITERATION NUMBERS IN EACH NEWTON-RAPHSON STEP WITH THE SAME PRECONDITIONER FOR EVERY STEP.
THE TEST CASE IS: IEEE – 300

NR-iteration	M_2	M_3	SPAI	ILU(0)	ILUT(0.1)	ILUT(0.01)
1	69	20	**F**	**F**	462	53
2	60	28	**F**	**F**	**F**	85
3	59	35	**F**	**F**	339	105
4	121	53	**F**	**F**	**F**	156
5	139	74	**F**	**F**	**F**	198

The effectiveness of the proposed preconditioners can be also observed from the Tables III, IV, V and VI. In tables, **F** shows the failure. If the same preconditioner is used in all Newton-Raphson steps, iteration numbers are not changed significantly. The main reason for this is the eigenvalue based design of the proposed preconditioners. In the literature, it is proved that, eigenvalues of the Jacobian does not widely change for power flow analysis problems. So MSF based preconditioners can be computed only once and then the same preconditioner can be used in each steps of Newton Raphson iteration.

III. Conclusion

In this study, we presented a new preconditioner design for the iterative solution of the linear equation systems arising from the power flow simulations for electrical power networks. Although direct methods with sparse techniques are very common in the area of power system simulation, these types of methods are not suitable for large problems due to fill-in and, more importantly, provides limited parallelism. Therefore, iterative methods have to be considered in the area of power system simulations. Iterative methods, on the other hand, need preconditioners to accelerate the convergence. A new preconditioner based on the Matrix Sign Function (MSF) is presented in this work. The main idea is to remove the extreme eigenvalues to reduce the number of eigenvalue clusters. To do this, spectral division properties of the MSF is employed. The main computational cost of the preconditioner is the computation of MSF. Meanwhile some well-known computational tools like QR decomposition also increase the computational cost of the design. On the other hand, electrical power network matrices have extremely sparse structures. So based on this information, SPAI (Sparse Approximate Inverse) method can be employed to produce the MSF. With this approach, sparsity structure of the Jacobian is preserved. Our method uses sparse QR factorization which is computationally expensive. This additional cost, however, can be amortized by the fact that it will be computed only once and the same preconditioner can be used effectively in subsequent Newton-Raphson iterations. In this regard, our algorithm has an advantage over the well-known preconditioner, such as incomplete LU. Computational tools and the structure of the suggested preconditioner are suitable for parallel processing. So our foresight about the parallel implementation of the suggested preconditioners will be effective and reliable. In our future work, we plan to improve the computational efficiency of the method and implement it on a parallel platform to solve larger problems.

IV. References

[1] H. Dag, A. Semlyen, A New Preconditioned Conjugate Gradient Power Flow , *IEEE Trans. Power Systems, Vol:18,* No:4, 2003, pp:1248- 1255

[2] L. Boumediena, M. Khiat, M. Rahli, A. Chaker, Harmonic Power Flow in Electric Power Systems Using Modified Newton Raphson Method, *International Review of Electrical Engineering, Vol:4* No:3, 2009, pp:410-416.

[3] A. Rashidinejad, M. R. Habibi , H. Khorasani, M. Rashidinejad, A Combinatorial Approach for Active and Reactive Power Flow Tracking, *International Review of Electrical Engineerin, Vol:5,* No:5, 2010,pp: 2209-2216.

[4] M. Hojabri, H. Hizam, N. Mariun, S. M. Abdullah, E. Pouresmaeil, Available Transfer Capability for Power System Planning Using Krylov-Algebraic Method, *International Review of Electrical Engineerin, Vol:5,* No:5, 2010,pp: 2251-2262.

[5] R. Jahani, H. C. Nejad, A. S. Malekshah, H. A. Shayanfar, *International Review of Electrical Engineering, Vol:5,* No:6, 2010,pp: 2786-2794.

[6] G. Golub, C. Van Loan, *Matrix Computations (* The John Hopkins University Press, 1996).

[7] R. Freund, G. Golub, N. Nachtigal, Iterative Solution of Linear Systems, *Acta Numerica, Cambridge University Press,* 1991,pp:57-100

[8] H. Dag, F. L. Alvarado, Computation-free Preconditioners for the Parallel Solution of the Power System Problems, *IEEE Trans. Power Systems, Vol:12,* No:2, 1997, pp:585-591.

[9] K. Chen, *Matrix Preconditioning Techniques and Applications* (Cambridge University Press, 2005).

[10] M. Benzi, Preconditioning Techniques for Large Linear Systems: A Survey", *Journal of Computational Pyhsics, Vol:182*, Issue:2, 2002, pp:418-477.

[11] W. W. Hager, *Applied Numerical Linear Algebra* (Prentice-Hall Publishing, USA, 1988).

[12] N. J. Higham, *Functions of Matrices, Theory and Computation* (SIAM Publishing, 2008).

[13] M. J. Grote, T. Huckle, Parallel Preocnditioning with Sparse Approximate Inverses, *SIAM J. of Scient. Comput. Vol 18* (3), 1997, pp:838-853.

[14] Z. Bai, J. Demmel, Design of a Parallel Non-symmetric Eigenroutine Toolbox, Part 1, *Technical Report, UCB/CSD-92-718, UC Berkeley,* 1992.

[15] C. S. Kenney, A. J. Laub, The Matrix Sign Function, *IEEE Trans. On Automatic Control, Vol:40* No:8, 1995, pp:1330-1348.

[16] C. Kenney, A. J. Laub, Rational Iteration Methods for the Matrix Sign Function, *SIAM J. on Math. Anal. Appl. Vol:21* , 1991, pp: 487-494.

[17] C. Kenney, A.J. Laub, On Scaling Newton's Method for Polar Decomposition and the Matrix Sign Function, *SIAM J. on Math. Anal. Appl. Vol:13*(3) , 1992, pp:688-706.

[18] Ç. K Koç. , B. Bakkaloğlu, L. S. Seigh, Computation of Matrix Sign Function Using Continued Fraction Expansion, *IEEE Trans. On Automatic Control, Vol:39*, No:8, 1994.

[19] S. T. Barnard, M. J. Grote, A Block Version of the SPAI Preconditioner, *9th SIAM Conf. on Parall. Process. For Sci. Comp.*, March 22-24, 1999, San Antonio, Texas, USA.

[20] E. F. Yetkin, H. Dag, "Applications of Eigenvalue Inclusion Theorems in Model Order Reduction", *Scientific Computing in Electrical Engineering, SCEE, 2008*, Espoo, Finland.

[21] Y. Saad, *Iterative Methods for Linear Sparse Systems, Second Edition* (SIAM, USA, 2003).

[22] H. Dağ, E. F. Yetkin, A Spectral Divide and Conquer Method Based Preconditoner Design for Power Flow Analysis, *Int. Conf. on Power System Technology (POWERCON)*,2010, Hangzhou, P. R. of China.

[23] R. D. Zimmerman, C. E. Murillo-Sánchez, and R. J. Thomas, MATPOWER's Extensible Optimal Power Flow Architecture, *IEEE Power and Energy Society General Meeting*, July, 26-30 2009, Calgary, Canada.

[24] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. Van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition* (SIAM Publishing, 1994).

Authors' information

[1]Information Technologies Department, Kadir Has University, Istanbul, Turkey. Email: hasan.dag@khas.edu.tr
[2]Informatics Institute, Istanbul Technical University, Istanbul, Turkey. Email: e.fatih.yetkin@be.itu.edu.tr
[3]Department of Computer Engineering, Middle East Technical University, Ankara, Turkey. Email: manguoglu@ceng.metu.edu.tr

Hasan Dağ received the B.Sc. degree in Electrical Engineering from Istanbul Technical University, Turkey and the M. Sc. and Ph. D. degrees in Electrical and Computer Engineering from the University of Wisconsin-Madison. Currently, he is a professor at the Information Technologies Department, Kadir Has University, Turkey. He is also a board member of the National Center for High Performance Computing of Turkey. His main interests include the analysis of large scale power systems, the use of iterative methods and parallel computation.

E. Fatih Yetkin received the B.Sc. degree in Electronics Engineering from Uludağ University Turkey and M.Sc. in Computational Science and Engineering from Istanbul Technical University, Turkey. He is currently a Ph.D. student in Computational Science and Engineering program in the same university. His main research interests are model order reduction, iterative methods and parallel computing.

Murat Manguoğlu received his B.sc. degree in Electrical and Electronics Engineering from Middle East Technical University, Ankara, Turkey, M.Sc. in Computational Engineering and Science from the University of Utah, Salt Lake City, Utah, USA, and Ph.D. degree in Computer Science from Purdue University, West Lafayette, IN, USA. His main research interests are parallel numerical linear algebra, sparse linear algebra, and high performance computing.